# AWS CLI – H3 Storage

**Creating a Bucket**

*aws s3 mb s3://<unique bucket name>*

**Configuring bucket with specs**

*aws s3api put-bucket-acl --bucket my-unique-bucket-name --acl private*

setting read permissions for profile esthry

*aws s3api put-bucket-acl --bucket <bucket-name> --acl public-read --profile esthry*

*Note* that if the region is not specified, the default region configured in the AWS CLI will be used.

**Create a logs directory in bucket**

*aws s3api put-object --bucket my-unique-bucket-name --key logs/*

**Upload files to bucket from server**

1. Make sure EC2 instance has AWS CLI installed

2. Copy credentials from Learner Lab to instance credentials (including session token!)

*aws s3 cp /path/to/local/file s3://your-bucket-name/logs/filename*

**Delet object from S3 bucket**

*aws s3 rm s3://my-bucket/my-directory/my-object*

**Create a new dir and sync this folder with docker mounted volume**

*aws s3 sync ~/init.sql/ s3://my-unique-bucket-6940/backend-data --acl public-read*

**Create website (this doesn’t fully work yet unfortunately)  
  
<!**DOCTYPE html>

<html>

<head>

<!-- head definitions go here -->

</head>

<body>

<!-- the content goes here -->

</body>

</html>

*aws s3 website s3://<bucket-name> --index-document index.html*

*aws s3api put-object-acl --bucket <bucket-name> --key index.html --acl public-read*

access via

*http://<bucket-name>.s3-website-us-east-1.amazonaws.com*

**List user groups from IAM**

*aws iam list-groups*

**Create a group**

*aws iam create-group --group-name developers*

**Attach policies**

*aws iam attach-group-policy --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess --group-name developers*

**List user ARN to configure user policy**

*aws sts get-caller-identity*

*<*copy ARN>

**EFS**

*aws efs create-file-system --creation-token <your-creation-token>*

*aws efs create-mount-target --file-system-id <your-file-system-id> --subnet-id <your-subnet-id> --security-group <your-security-group-id>*

**Create new Volume (EBS: 16gb gp2)**

Find availability zone

*aws ec2 describe-instances --instance-ids <instance-id> --query 'Reservations[\*].Instances[\*].Placement.AvailabilityZone' --output text*

Create volume

*aws ec2 create-volume --size 16 --volume-type gp2 --availability-zone us-east-1d*

Attach volume *aws ec2 attach-volume --volume-id <volume-id> --instance-id <instance-id> --device /dev/sdf*

List volumes in case you need id

aws ec2 describe-volumes

**Create Snapshot of Volume**

*aws ec2 create-snapshot --volume-id <volume-id> --description "Snapshot of 16GB Volume" --tag-specifications 'ResourceType=snapshot,Tags=[{Key=backupTag,Value=FirstBackup}]'*

List snapshots you created

*aws ec2 describe-snapshots --owner-ids self*

**Example of a Resource Policy from Policy Generator**(to give jane.smith permissions for specific s3 bucket)

{

"Id": "Policy1679860899092",

"Version": "2012-10-17",

"Statement": [

{

"Sid": "Stmt1679860890098",

"Action": "s3:\*",

"Effect": "Allow",

"Resource": "arn:aws:s3:::lab-resources-ee1e4137-89e0-10dc-2d94-286b988bb680",

"Principal": {

"AWS": [

"arn:aws:iam::517058338644:user/jane.smith"

]

}

}

]

}

**Add Resource Policy to Bucket Policy**

*aws s3api put-bucket-policy --bucket lab-resources-ee1e4137-89e0-10dc-2d94-286b988bb680 --policy* ‘{

"Id": "Policy1679860899092",

"Version": "2012-10-17",

"Statement": [

{

"Sid": "Stmt1679860890098",

"Action": "s3:\*",

"Effect": "Allow",

"Resource": "arn:aws:s3:::lab-resources-ee1e4137-89e0-10dc-2d94-286b988bb680",

"Principal": {

"AWS": [

"arn:aws:iam::517058338644:user/jane.smith"

]

}

}

]

}’

**Block Public Access to S3 bucket**

*aws s3api put-public-access-block --bucket <bucket-name> --public-access-block-configuration "BlockPublicAcls=true,IgnorePublicAcls=true,BlockPublicPolicy=true,RestrictPublicBuckets=true"*

**Allow access to specific Object**

,{

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::lab-resources-<id>/public\_data/public-access-doc-02.txt" } (…]})

**Create encrypted key**(for user alice.ame with symmetric key type)

*aws kms create-key --description "Symmetric key for alice.acme" --policy "{\"Version\":\"2012-10-17\",\"Id\":\"key-policy\",\"Statement\":[{\"Sid\":\"Enable IAM User Permissions\",\"Effect\":\"Allow\",\"Principal\":{\"AWS\":\"arn:aws:iam::123456789012:user/alice.acme\"},\"Action\":\"kms:\*\",\"Resource\":\"\*\"}]}" --tags KeyOwner=alice.acme*

**List Keys**

*aws kms list-keys*

**Server side encryption on specific file**

*aws s3 cp s3://my-bucket/my-file.txt s3://my-bucket/my-file.txt --sse-customer-algorithm AES256 --sse-customer-key my-encrypted-key*

**Create and configure an EFS File System with json**

a*ws efs create-file-system --creation-token MyEfsFileSystem --performance-mode generalPurpose --encrypted --kms-key-id <kms-key-id> --throughput-mode bursting --tags Key=Name,Value=MyEfsFileSystem --region <region> --policy* ‘{

"Version": "2012-10-17",

"Id": "efs-policy-wizard-6b84358f-dab5-487f-9f85-87c5bbac3996",

"Statement": [

{

"Sid": "efs-statement-6278c195-eedb-4e0a-a901-5098a1ee6b27",

"Effect": "Allow",

"Principal": {

"AWS": "\*"

},

"Action": [

"elasticfilesystem:ClientMount"

],

"Condition": {

"Bool": {

"elasticfilesystem:AccessedViaMountTarget": "true"

}

}

},

{

"Sid": "efs-statement-1d0f83e2-852f-4a40-a23a-eec74a6f8221",

"Effect": "Deny",

"Principal": {

"AWS": "\*"

},

"Action": "\*",

"Condition": {

"Bool": {

"aws:SecureTransport": "false"

}

}

}

]

}’